**Amendments to the Specification:**

Please amend paragraph [0017] as shown below.

[0017] The process to implement the buffer overrun is commonly known as "smashing the stack." ~~It is detailed in technical depth in "Smashing the Stack for Fun and Profit," available on the Internet at www.phrack.com/archive.~~ An exemplary process for smashing the stack or overrunning buffers is illustrated in Figures 1A and 1B. Program 10 includes the function "MAIN" which defines array variable 12 (illustrated in Figure 1B) with the label "LARGE." Array LARGE is defined with a length of two thousand bytes. After creating the array LARGE, function MAIN fills it with two thousands "X" characters, as shown in Figure 1B. Next, function MAIN calls the function "OVERFLOW" with a pointer to array "LARGE" passed as an argument. In OVERFLOW, array variable 14, labeled "SMALL," is defined with a length of one hundred bytes (illustrated in Figure 1C). The left side of Figure 1D shows program stack 20 illustrating how memory is allocated when the OVERFLOW function is called in program 10. As shown in Figure 1D, the array SMALL is allocated one hundred bytes, represented by block 22. After SMALL, program stack 20 has memory reserved for the stack frame pointer (SFP) in block 24, the return instruction pointer (IP) in block 26, and the pointer (in block 28) that was pushed onto the stack when OVERFLOW was called. After creating array SMALL, the OVERFLOW function simply copies the contents of the array LARGE into the memory reserved for array SMALL using C's *strcpy* function.